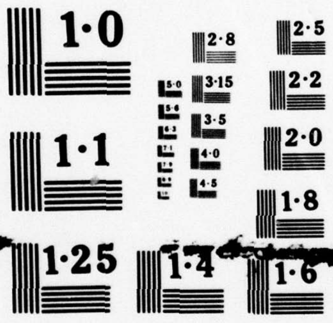
AD-A060 674 ARMY ARMAMENT RESEARCH AND DEVELOPMENT COMMAND ABERD--ETC F/G 9/2
THE DAVE SYSTEM: A CRITIQUE AND GUIDE FOR USE AT THE BALLISTIC --ETC(U)
AUG 78 M A HIRSCHBERG, J LACETERA
UNCLASSIFIED ARBRL-MR-02858 SBIE-AD-E430 107 NL

1 OF 1
ADA
060674

END
DATE
FILMED
1 -79
DDC



NATIONAL BUREAU OF STANDARDS
MICROCOPY RESOLUTION TEST CHART

LEVEL II

12 B.S.

AD-E430 107

ADA060674

DDC FILE COPY

MEMORANDUM REPORT ARBRL-MR-02858

THE DAVE SYSTEM: A CRITIQUE AND GUIDE FOR USE AT THE BALLISTIC RESEARCH LABORATORY

Morton A. Hirschberg
Joseph Lacetera
William Buchheister

DDC
RECEIVED
NOV 1 1978
B

August 1978

US ARMY ARMAMENT RESEARCH AND DEVELOPMENT COMMAND
BALLISTIC RESEARCH LABORATORY
ABERDEEN PROVING GROUND, MARYLAND

Approved for public release; distribution unlimited.

78 09 18 067

Destroy this report when it is no longer needed.
Do not return it to the originator.

Secondary distribution of this report by originating or sponsoring activity is prohibited.

Additional copies of this report may be obtained from the National Technical Information Service, U.S. Department of Commerce, Springfield, Virginia 22161.

The findings in this report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

The use of trade names or manufacturers' names in this report does not constitute indorsement of any commercial product.

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER (9) MEMORANDUM REPORT (14) ARBRL-MR-02858	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) (6) THE DAVE SYSTEM: A CRITIQUE AND GUIDE FOR USE AT THE BALLISTIC RESEARCH LABORATORY		5. TYPE OF REPORT & PERIOD COVERED
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) (10) Morton A. Hirschberg, Joseph Lacetera, William Buchheister		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS US Army Ballistic Research Laboratory (ATTN: DRDAR-BLB) Aberdeen Proving Ground, MD 21005		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS (16) 1L161102AH43
11. CONTROLLING OFFICE NAME AND ADDRESS US Army Armament Research & Development Command US Army Ballistic Research Laboratory (ATTN: DRDAR-BL) Aberdeen Proving Ground, MD 21005		12. REPORT DATE (11) AUG[illegible] [illegible]78
		13. NUMBER OF PAGES 37
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) (18) SBIE (12) 34p.		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

(19) AD-E430 107

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Program Testing	Debugging
Program Validation	Static Analysis
Program Error Detection	Software Tools
Software Validation	

20. ABSTRACT (Continue on reverse side if necessary and identify by block number) (fek) DAVE is a static program analyzer. That is, DAVE looks at ANSI standard FORTRAN programs of moderate size and provides documentation, analysis, validation and error detection (hence, its well hidden acronym), without executing the program. This paper endeavors to critique the DAVE system in an unbiased fashion. The authors were without prior knowledge of the system before its purchase. It is the authors' conclusion; however, that DAVE represents a useful validation tool for analysis of FORTRAN programs, beyond that initially provided by compiler.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

78 09 18 06[illegible]

393471

TABLE OF CONTENTS

ACCESSION for

NTIS White Section ✓

DDC Buff Section □

UNANNOUNCED □

JUSTIFICATION

BY

DISTRIBUTION/AVAILABILITY CODES

Dist. AVAIL. and/or SPECIAL

A

PRECEDING PAGE BLANK-NOT FILMED

I. INTRODUCTION

Dave[1] is one of several program validation techniques in use for the analysis of FORTRAN programs. The need for techniques which enable programmers to validate programs and reduce costs is well documented[2,3].

DAVE is a static analyzer. That is, DAVE looks at FORTRAN programs and provides error analysis and documentation of code without executing the program. While this technique is not foolproof, it presents the user considerable analysis of FORTRAN programs above that provided by FORTRAN compilers. A comparison of a "clean" FORTRAN program compiled on a CDC Cyber 173 and analyzed by DAVE is shown in the appendices (Appendix A and B).

Fairley[4], in a recent article, listed ten items of information which can be obtained by static analysis. These are quoted verbatim below:

> *"The information that can be obtained by static analysis includes (1) syntactic error messages; (2) number of occurrences of source statements by type; (3) cross-reference maps of identifier usage; (4) analysis of how the identifiers are used in each statement (data source, data sink, calling parameter, dummy parameter, subscript, etc.); (5) subroutines and functions called by each routine; (6) uninitialized variables; (7) variables set but not used; (8) isolated code segments that cannot be executed under any set of input data; (9) departures from coding standards (both language standards and local practice standards); and (10) misuses of global variables, common variables, and parameter lists (incorrect number of parameters, mismatched types, uninitialized input parameters, output parameters not assigned to, output parameters assigned to but never used, parameters never used for either input or output, etc.".*

[1]Osterweil, L. J., Fosdick, L. D., "DAVE A Validation, Error Detection and Documentation System for FORTRAN Programs", Department of Computer Science, University of Colorado, TR #CU-CS-071-75, February 1975.

[2]Elspas, B., Levitt, K. N., Waldinger, J. and Waksman, A., "An Assessment of Techniques for Proving Program Correctness", ACM Computing Survey, 4, pp. 97-147, June 1972.

[3]Boehm, B., "Software and Its Impact: A Quantitative Assessment", Datamation, 14, 5, pp 48-59, May 1973.

[4]Fairley, R. E., "Tutorial: Static Analysis and Dynamic Testing of Computer Software", Computer, pp. 14-23, April 1978.

PRECEDING PAGE BLANK-NOT FILMED

Prior to using DAVE, one author had some experience with RXVP[5]. Outside of that, the authors had no prior experience with validating techniques other than that provided by standard FORTRAN compilers. We feel, therefore, that we can present an unbiased evaluation of DAVE.

II. DAVE AND CRITIQUE OF DAVE

The DAVE system is well documented in Reference 1; however, a few of its characteristics and highlights will be presented.

DAVE is written in ANSI FORTRAN and contains approximately 20,000 source statements. DAVE consists of a number of files which can be and have been combined to form a procedure (see Section III). The most important of these files for the user is the FORTRAN program to be analyzed. This may consist of a single subprogram, a group of subprograms, or a main program and its subprograms.

DAVE attempts to locate violations of FORTRAN rules and thus detect the presence of common programming errors. DAVE produces 13 error messages (e.g., the number of arguments in the parameter lists do not correspond), 24 warning messages (e.g., a variable in a parameter list is used for neither input nor output), and 5 general messages (e.g., subprogram "NAME" is never called). The output from DAVE is extensive and a good portion of it is informative in nature and points to "errors" that may not affect the running of the program. Programmers who use sophisticated techniques (not all of which are allowed under all versions of FORTRAN compilers), may find a good number of warning errors after putting their programs through DAVE.

The authors have taken the simple input case provided with DAVE and cleaned up all the errors found by the CDC NOSBE FORTRAN compiler. These "clean" programs were then processed through DAVE which found numerous errors; errors serious enough to show that the programs were indeed not valid. Portions of these results are shown in the appendices.

As a tool, DAVE does perform many of the ten tasks outlined by Fairley. It does not provide a complete set of syntactic error messages but does provide many such messages. DAVE does not provide the number of occurrences of source statements by type, cross reference maps of identifier usage, or an analysis of how identifiers are used. DAVE does provide information as to all the other items listed by Fairley.

Finally, we feel that the limits of application of DAVE to FORTRAN programs should be mentioned here; and these are quoted verbatim below:

[5]Miller, E. F., Jr., "RXVP: An Automated Verification System for FORTRAN", in Proc. Computer Science and Statistics: 8th Annual Symposium on the Interface, Los Angeles, CA, February 1975.

"The limits on the size of the FORTRAN programs to be analyzed by DAVE are: a maximum of 100 subprograms run through together, each of which has no more than 500 blocks. All declarations are counted as one block; otherwise each statement is a block, with logical IF's counting as two blocks; COMMENTS and Formats are excluded from the count. Although the limit is 100 program units run together, the larger the group, the more strain placed on all internal arrays and overflow may occur. Detailed information on all size limitations will appear in the User Manual[6]".

Although our experience with DAVE is still limited, we hope that this document will encourage the use of DAVE so that we can gather further statistics and report upon these in the future.

III. INSTRUCTIONS ON THE USE OF DAVE AT THE BALLISTIC RESEARCH LABORATORY

This section deals with the mechanics of using DAVE; that is, which procedures must be used, what file DAVE resides on, and what the run stream for DAVE is.

For the convenience of users at the BRL we have set up a procedure file to execute the DAVE code. The following is a sample deck for its use:

```
BRL76, STMFZ,T40.
ACCOUNT (BRL76)
ATTACH (OLDPL, PLTESTFLC, ID = CMCWB)
UPDATE (F, C=TEST, D,8)
BEGIN, DAVE, DAVE, INPUT, TEST.
7/8/9
7/8/9
SI=ON
7/8/9
```

The ATTACH command accesses the permanent file PLTESTFLC and assigns it the local file name OLDPL. PLTESTFLC is an UPDATE program library which we wish to use as input to DAVE, and can have any name. In this particular example it contained our modified version of the test program which was provided with DAVE to exercise its capabilities. As noted earlier, this program compiled without errors on the 4.6(OPT=2)compiler.

The UPDATE command causes UPDATE to write the compile output decks on the file named TEST. It is a full update and the output has 80 data columns. This file is to be used as the input file by the DAVE procedure file, and must have the same name on both cards.

[6]"Installing DAVE on a Computer", University of Colorado, Undated Notes.

The FORTRAN program to be analyzed may consist of a single subprogram, several subprograms, a main program and subprograms, or just a main program.

Finally, the BEGIN card causes execution of the DAVE procedure file. Note that the file INPUT contains the record SI=ON which indicates that simulations of missing subprograms are desired. Another input option to DAVE uses the keyword SU which can be used to suppress the printing of errors, warnings, or messages.

If the test program were not on an UPDATE program library but on cards, it would have to follow the options cards in the input file, and be separated from them by an end-of-file.

It should be noted that the dayfile in Appendix B shows the entire job control stream, generated by the "PROC", and that these instructions are not part of the job control deck.

Appendix A contains the listing of the FORTRAN compilation of the main program on the permanent file PLTESTFLC. It is free of error messages or diagnostic information of any type, as were all of the subroutines (not shown). Appendix B contains the output resulting from executing DAVE on this same main program. The output is extensive showing errors, warnings, and diagnostic messages. It is representative of the output resulting from DAVE operating on the subroutines. From this output it is apparent that one obvious aspect of DAVE's superiority over FORTRAN compilers is its ability to analyze coding in the context of subroutines and functions called by a program or subprograms.

IV. SUMMARY

Based on limited use, DAVE does everything it is purported to do. From the standpoint of economics, DAVE is inexpensive to purchase, and in an era of expanding work and a decreasing work force, a worthwhile error detector.

Other new tools being developed for the analysis of FORTRAN programs are either not yet available for general consumption[7], or require additional resources for use[8]. In the latter case, not everyone can afford

[7]Clarke, L., "A System to Generate Test Data and Symbolically Execute Programs", Department of Computer Science, University of Colorado, #CU-CS-060-75, February 1975.

[8]Browne, J. C. and Johnson, D. B., "FAST: A Second Generation Program Analysis System", Proceedings of 3rd International Conference on Software Engineering, IEEE Catalog No. 78CH1317-7C, pp. 142-148, May 1978.

the lease or purchase of a data base analysis system (in one case, System 2000). In the former case, we are awaiting the use of a symbolic evaluator.

REFERENCES

1. Osterweil, L. J., Fosdick, L. D., "DAVE - A Validation, Error Detection and Documentation System for FORTRAN Programs", Department of Computer Science, University of Colorado, TR #CU-CS-071-75, February 1975.

2. Elspas, B., Levitt, K. N., Waldinger, J. and Waksman, A., "An Assessment of Techniques for Proving Program Correctness", ACM Computing Survey, 4, pp. 97-147, June 1972.

3. Boehm, B., "Software and Its Impact: A Quantitative Assessment", Datamation, 14, 5, pp. 48-59, May 1973.

4. Fairley, R. E., "Tutorial: Static Analysis and Dynamic Testing of Computer Software", Computer, pp. 14-23, April 1978.

5. Miller, E. F., Jr., "RXVP: An Automated Verification System for FORTRAN", in Proc. Computer Science and Statistics: 8th Annual Symposium on the Interface, Los Angeles, CA, February 1975.

6. "Installing DAVE on a Computer", University of Colorado, Undated Notes.

7. Clarke, L., "A System to Generate Test Data and Symbolically Execute Programs", Department of Computer Science, University of Colorado, "CU-CS-060-75, February 1975.

8. Browne, J. C. and Johnson, D. B., "FAST: A Second Generation Program Analysis System", Proceedings of 3rd International Conference on Software Engineering", IEEE Catalog No. 78CH1317-7C, pp. 142-148, May 1978.

PRECEDING PAGE BLANK-NOT FILMED

APPENDIX A

FORTRAN COMPILATION OF TEST PROGRAM
WHICH SHOWS NO ERRORS.

PRECEDING PAGE BLANK-NOT FILMED

```
              PROGRAM MAIN(INPUT,OUTPUT)
              COMMON/B1/CA1,BA
              COMMON/BLK1/CA,D218,Y228(6)
              EXTERNAL SUBEX
              DIMENSION XDAT(5),XDT(5,2)
              INTEGER I236
              DATA I220/1/ ,X221/1./
              LASRF(X,Y)=5.*E101(C)+X+Y
              C=1.
              B=1.
              D219=1.0+X221
              I236=1+I220
              L=1
              LOC=1
              M=1
              Y=1.0
              D=2.0
              DO 100 I1=1,5
              DO 200 J1=1,2
              XDT(I1,J1)=I1+J1
          200 CONTINUE
          100 CONTINUE
              IF(A.EQ.B) X = CA1
              X = M+CA
              B = E101(1.)
              CALL SUB103(3,B+C,Y+1)
              CALL SUB103(A,B+C,Y+1)
              CALL SUB105(SUBEX,3)
              CALL SUB106(SUBEX,3)
              CALL SUB208(A,CA)
              DO 10 I = 1 , 10
              K = LOC + 1
           10 CONTINUE
              K = I + 6
              CALL SUB(1.,SUBEX)
              I = W201(CA)
              CALL SUB215(XDAT,B,C)
              CA = 1.
              CA = 2.
              BA=CA
              IF(D219.EQ.0) CA=3.+BA
              Y228(1)=6.+B
              X229=6.
              X230 = 1.
              IF(CA.EQ.L) X230=3.
              XDAT(5)=1.+X229+X230
              I = W201(X)+X
              I = W201(CA)+CA
              I = LASRF(2.,5.)+C
              I = FSIM(CA)
              CALL SUBSIM(X,A,B,D)
              STOP
              END
```

```
      PROGRAM MAIN       76/76    OPT=2 ROUND=+-*/                FTN 4.6+452         05/26/78  08.34.49         PAGE      2

         SYMBOLIC REFERENCE MAP (R=2)

   ENTRY POINTS       DEF LINE       REFERENCES
     44  MAIN               1

   VARIABLES      SN   TYPE               RELOCATION
     257  A            REAL                                REFS        23          27          30          51
     246  B            REAL                                REFS        23          26          27          37          51
                                                        DEFINED        10
       1  BA           REAL                  B1            REFS         2          41      DEFINED         40
     245  C            REAL                                REFS        26          27          37         2*49     DEFINED         9
       0  CA           REAL                  BLK1          REFS         3          24          30          36          40          45        2*48
                                                            50     DEFINED         38          39          41
       0  CA1          REAL                  B1            REFS         2          23
     254  D            REAL                                REFS        51      DEFINED         17
       1  D218         REAL                  BLK1          REFS         3
     247  D219         REAL                                REFS        41      DEFINED         11
     262  I            INTEGER                             REFS        34      DEFINED         31          36          47          48          49
                                                            50
     255  I1           INTEGER                             REFS       2*20     DEFINED         18
     233  I220         INTEGER                             REFS        12      DEFINED          7
     244  I236         INTEGER                             REFS         6      DEFINED         12
     256  J1           INTEGER                             REFS       2*20     DEFINED         19
     263  K            INTEGER                          DEFINED        32          34
     250  L            INTEGER                             REFS        45      DEFINED         13
     251  LOC          INTEGER                             REFS        32      DEFINED         14
     252  M            INTEGER                             REFS        24      DEFINED         15
     261  R            REAL                                REFS        42      DEFINED         25
     260  X            REAL                                REFS       2*47         51      DEFINED         23          24
     266  XDAT         REAL        ARRAY                   REFS         5          37      DEFINED         46
     273  XDT          REAL        ARRAY                   REFS         5      DEFINED         20
     234  X221         REAL                                REFS        11      DEFINED          7
     264  X229         REAL                                REFS        46      DEFINED         43
     265  X230         REAL                                REFS        46      DEFINED         44          45
     253  Y            REAL                                REFS        26          27      DEFINED         16
       2  Y228         REAL        ARRAY     BLK1          REFS         3      DEFINED         42

   FILE NAMES          MODE
       0  INPUT
      20  OUTPUT

   EXTERNALS          TYPE     ARGS          REFERENCES
          E101        REAL       1             25          49
          FSIM        REAL       1             50
          SUM                    2             35
          SUBFX                  0              4          28          29          35
          SUBSIM                 4             51
          SUB103                 3             26          27
          SUB105                 2             28
          SUB106                 2             29
          SUB208                 2             30
          SUB215                 3             37
          W201        REAL       1             36          47          48
          PROGRAM MAIN       76/76    OPT=2 ROUND=+-*/               FTN 4.6+452          05/26/78  08.34.49        PAGE      3

INLINE FUNCTIONS    TYPE     ARGS          DEF LINE  REFERENCES
      LASRF       INTEGER     2     SF        8           49

STATEMENT LABELS              DEF LINE    REFERENCES
     0  10                        33         31
     0  100                       22         18
     0  200                       21         19

LOOPS  LABEL      INDEX        FROM-TO      LENGTH      PROPERTIES
   55  100      * I1            18 22          6B                    NOT INNER
   56  200        J1            19 21          3B        INSTACK
  114  10         I             31 33          1B        INSTACK

COMMON BLOCKS    LENGTH
      B1              2
      BLK1            8

STATISTICS
  PROGRAM LENGTH                 245B        165
  BUFFER LENGTH                  40B          32
  SCM LABELED COMMON LENGTH       12B          10
        55000B SCM USED
```

APPENDIX B

RESULTS OF DAVE EXECUTION ON TEST PROGRAM

```
***  04/24/78  SCOPE 2.1.4  B R L  VER 004  ***   05/26/78      78146

SYS DEVICES 819/ 4/PE FLS=377K FLL=1750K MXS=240K MXL=1300K MXB=1300B

HH.MM.SS CPU SECOND ORIGIN
         08.39.42.MFA. AF    BRL NOS/BE 1.2 L447 VERSION 4.1 05/15/78
08.40.11 00000.002 MFZ.      -SBL48,STMFZ,T40.   EXECUTE DAVE PROC
08.40.11 00000.003 JOB.      -ACCOUNT(BD***)
08.40.12 00000.025 JOB.      -ATTACH(OLDPL,PLTESTFLC,ID=SBLJL)
08.40.12 00000.028 MFZ.         PF254 - CYCLE    2 ATTACHED FROM SN=SYSTEM
08.40.12 00000.029 LOD.      -UPDATE(F,C=TEST,D,8)
08.40.14 00000.077 USR.       UPDATE COMPLETED
08.40.14 00000.078 LOD.      -BEGIN,DAVE,DAVE,INPUT,TEST.
08.40.14 00000.091 MFZ.         PF646 - PFMACRO - ATTACH  - DAVE    - DAVE
08.40.14 00000.097 MFZ.         PF254 - CYCLE    1 ATTACHED FROM SN=SYSTEM
08.40.15 00000.119 JOB.      -MAP(OFF)
08.40.15 00000.120 LOD.      -FTN(I=ZZCCLAA,B=COMP,L=0)
08.40.16 00000.180 USR.            .054 CP SECONDS COMPILATION TIME
08.40.16 00000.185 JOB.      -ATTACH(PH0B,ID=SBLJL)
08.40.16 00000.185 MFZ.         PF053 - LFN  IS   PH0B
08.40.16 00000.189 MFZ.         PF254 - CYCLE    2 ATTACHED FROM SN=SYSTEM
08.40.16 00000.189 JOB.      -LOAD(COMP)
08.40.16 00000.192 LOD.      -PH0B(INPUT,TEST)
08.40.17 00000.334 MFZ.         LD610 - FLS REQUIRED TO LOAD - 0012275 OU.COG
08.40.17 00000.335 MFZ.         LD603 - EXECUTION INITIATED OS.EXP
08.40.17 00000.335 USR.        FORTRAN LIBRARY 452            08/04/77
08.40.17 00000.459 USR.           STOP
08.40.17 00000.459 USR.             .123 CP SECONDS EXECUTION TIME
08.40.17 00000.464 JOB.      -ATTACH(PH1B,ID=SBLJL)
08.40.17 00000.464 MFZ.         PF053 - LFN  IS   PH1B
08.40.18 00000.468 MFZ.         PF254 - CYCLE    2 ATTACHED FROM SN=SYSTEM
08.40.18 00000.469 JOB.      -ATTACH(COMDAT,ID=SBLJL)
08.40.18 00000.469 MFZ.         PF053 - LFN  IS   COMDAT
08.40.18 00000.473 MFZ.         PF254 - CYCLE    3 ATTACHED FROM SN=SYSTEM
08.40.18 00000.474 JOB.      -ATTACH(DBLIB,ID=SBLJL)
08.40.18 00000.474 MFZ.         PF053 - LFN  IS   DBLIB
08.40.18 00000.478 MFZ.         PF254 - CYCLE    1 ATTACHED FROM SN=SYSTEM
08.40.18 00000.478 JOB.      -LIBRARY(DBLIB)
08.40.18 00000.482 LOD.      -PH1B.
08.40.19 00001.031 MFZ.         LD610 - FLS REQUIRED TO LOAD - 0020655 OU.COG
08.40.19 00001.032 MFZ.         LD603 - EXECUTION INITIATED OS.EXP
08.40.19 00001.032 USR.        FORTRAN LIBRARY 452            08/04/77
08.40.24 00002.483 USR.           STOP
08.40.24 00002.483 USR.            1.449 CP SECONDS EXECUTION TIME
08.40.24 00002.484 LOD.      -RETURN(PH1B)
08.40.24 00002.495 JOB.      -ATTACH(PH2B,ID=SBLJL)
08.40.24 00002.496 MFZ.         PF053 - LFN  IS   PH2B
08.40.25 00002.499 MFZ.         PF254 - CYCLE    1 ATTACHED FROM SN=SYSTEM
08.40.25 00002.500 LOD.      -PH2B.
08.40.26 00003.012 MFZ.         LD610 - FLS REQUIRED TO LOAD - 0021562 OU.COG
08.40.26 00003.015 MFZ.         LD603 - EXECUTION INITIATED OS.EXP
08.40.26 00003.015 USR.        FORTRAN LIBRARY 452            08/04/77
08.40.36 00005.575 USR.           STOP
08.40.36 00005.575 USR.            2.557 CP SECONDS EXECUTION TIME
08.40.36 00005.576 LOD.      -RETURN(PH2B)
08.40.36 00005.588 JOB.      -ATTACH(PH3B,ID=SBLJL)
08.40.36 00005.588 MFZ.         PF053 - LFN  IS   PH3B
08.40.37 00005.592 MFZ.         PF254 - CYCLE    1 ATTACHED FROM SN=SYSTEM
08.40.37 00005.592 LOD.      -PH3B.
08.40.38 00005.875 MFZ.         LD610 - FLS REQUIRED TO LOAD - 0015064 OU.COG
08.40.38 00005.877 MFZ.         LD603 - EXECUTION INITIATED OS.EXP
08.40.38 00005.877 USR.        FORTRAN LIBRARY 452            08/04/77
```

```
08.40.40 00006.815 USR.          STOP


08.40.40 00006.815 USR.               .435 CP SECONDS EXECUTION TIME
08.40.40 00006.815 LOD.        -RETURN(RM3R)
08.40.40 00006.823 LOD.        -REVERT.
08.40.41 00006.838 MFZ.           RM770 - MAXIMUM ACTIVE FILES              11
08.40.41 00006.839 MFZ.           RM771 - OPEN/CLOSE CALLS                 129
08.40.41 00006.839 MFZ.           RM772 - DATA TRANSFER CALLS            3,988
08.40.41 00006.839 MFZ.           RM773 - CONTROL/POSITIONING CALLS        131
08.40.41 00006.839 MFZ.           RM774 - RM DATA TRANSFER CALLS         2,321
08.40.41 00006.839 MFZ.           RM775 - RM CONTROL/POSITIONING CALLS     273
08.40.41 00006.839 MFZ.           RM776 - QUEUE MANAGER CALLS              448
08.40.41 00006.840 MFZ.           RM777 - RECALL CALLS                     356
08.40.41 00006.840 MFZ.           SCM             201.028  KWS
08.40.41 00006.840 MFZ.           LCM             174.415  KWS
08.40.41 00006.840 MFZ.           I/O               0.414  MW
08.40.41 00006.840 MFZ.           RMS               0.782  MWS
08.40.41 00006.841 MFZ.           USER              4.538  SEC
08.40.41 00006.841 MFZ.           JOB               6.843  SEC
08.40.41 00006.841 MFZ.           DIO           1 030.774  KW
08.40.41 00006.841 MFZ.           SC050 - 000001 SC/LC SWAPS

  ***********    SHL4824   0002039 LINES PRINTED
```

```
                         DAVE LEVEL 8.0



        ************************************************************
        *                                                          *
        *             DAVE  TERMINATION  NORMAL                    *
        *                                                          *
        ************************************************************



  ....................................................................
  .                                                                  .
  .   NOTE -- FOR MISSING SUBPROGRAMS THE FOLLOWING I/O BEHAVIOR     .
  .           HAS BEEN SIMULATED.                                    .
  .           A.  FOR FUNCTION SUBPROGRAMS, THE FUNCTION NAME HAS    .
  .               BEEN CLASSIFIED AS STRICT OUTPUT AND ALL ARGU-     .
  .               MENTS AS STRICT INPUT, NON-OUTPUT.                 .
  .           B.  FOR SUBROUTINE SUBPROGRAMS, ALL ARGUMENTS HAVE     .
  .               BEEN CLASSIFIED AS STRICT INPUT, NON-OUTPUT.       .
  .                                                                  .
  .           A SIMULATED SUBPROGRAM IS ASSUMED TO USE NO COMMON     .
  .           VARIABLES.  THE NUMBER AND DIMENSIONS OF ITS DUMMY     .
  .           ARGUMENTS HAVE BEEN INFERRED FROM THE FIRST INVO-      .
  .           CATION OF THE SUBPROGRAM BY THE PROGRAM UNIT           .
  .           INDICATED BELOW.                                       .
  .                                                                  .
  .           SIMULATED SUBPROGRAM           CALLER                  .
  .           --------------------           ------                  .
  .               ---*FSIM*--               -*SYSMAIN*-              .
  .               --*SUBSIM*-               -*SYSMAIN*-              .
  .                                                                  .
  ....................................................................



                    USER OPTIONS SPECIFIED THIS RUN
                    -------------------------------


  1.  SIMULATE I/O BEHAVIOR FOR MISSING SUBPROGRAMS (SI= ON).

  2.  RE-START OF PREVIOUS RUN   (RE=OFF).

  3.  SUPPRESS DIAGNOSTICS (SU=OFF).
```

```
                  DIAGNOSTIC SUMMARY -- PART 1
                  ----------------------------

                                 FREQUENCY

      SUBPROGRAM        ERRORS        WARNINGS       MESSAGES
      ----------        ------        --------       --------

         SYSMAIN            18              42              5
         BLKDATA                                            1
            E101             1               4              1
          SUB103                             2              2
          SUB302             1               6              2
          SUB105                                            1
          SUB106             1               1              2
          SUB208                                            1
            W201                             3              1
          SUB215                             1              1
             SUB                             4              1
             FUN                             1              1
            FSIM                                            1
          SUBSIM                                            1


                  DIAGNOSTIC SUMMARY -- PART 2
                  ----------------------------

          ERRORS                     WARNINGS                     MESSAGES

IDENT.NO.    FREQUENCY     IDENT.NO.    FREQUENCY     IDENT.NO.    FREQUENCY
---------    ---------     ---------    ---------     ---------    ---------

   101           1            201           1            301           2
   103           5            202           1            302           3
   105           1            203           2            303           2
   106           2            204           5            304          14
   108           2            205           1
   109           2            206           1
   110           4            208           1
   111           2            209           1
   112           2            210           3
                              211           1
                              213           2
                              214           3
                              215           1
                              216           5
                              217           2
                              218           1
                              219           1
                              220           1
                              221           1
                              222           1
                              223           2
                              224           1
                              225           1
                              226           1


                              227           1
                              228           1
                              229           8
                              230           3
                              231           2
                              232           2
                              233           2
                              234           1
                              235           1
                              237           3
```

```
                   CALL GRAPH
                   ----------

   SUBPROGRAM        CALLED BY         CALLS
   ----------        ---------         -----

    SYSMAIN                             E101
                                      SUB103
                                      SUB105
                                      SUB106
                                      SUB208
                                        W201
                                      SUB215
                                         SUB
                                        FSIM
                                      SUBSIM

----------------------------------------------------
          E101        SYSMAIN
                       SUB106

----------------------------------------------------
        SUB103        SYSMAIN           SUB302
----------------------------------------------------
        SUB302         SUB103           SUB106
----------------------------------------------------
        SUB105        SYSMAIN           SUB106
----------------------------------------------------
        SUB106        SYSMAIN             E101
                       SUB302
                       SUB105

----------------------------------------------------
        SUB208        SYSMAIN
----------------------------------------------------
          W201        SYSMAIN
----------------------------------------------------
        SUB215        SYSMAIN
----------------------------------------------------
           SUB        SYSMAIN
----------------------------------------------------
           FUN
----------------------------------------------------
          FSIM        SYSMAIN
----------------------------------------------------
        SUBSIM        SYSMAIN
----------------------------------------------------
```

SOURCE PROGRAM LISTING

$ IN THE CONTINUATION FIELD INDICATES THE EXPANSION
OF THE LOGICAL IF STATEMENT ON THE PREVIOUS LINE

```
BLOCK                    SOURCE

  1          PROGRAM MAIN(INPUT,OUTPUT)
  1          COMMON/B1/CA1,BA
  1          COMMON/BLK1/CA,D218,Y228(6)
  1          EXTERNAL SUBEX
  1          DIMENSION XDAT(5),XDT(5,2)
  1          INTEGER I236
  1          DATA I220/1/ ,X221/1./
  1          LASBF(X,Y)=5.*F101(C)+X+Y
  2          C=1.
  3          B=1.
  4          D219=1.0+X221
  5          I236=1+I220
  6          L=1
  7          LOC=1
  8          M=1
  9          Y=1.0
 10          D=2.0
 11          DO 100 I1=1,5
 12          DO 200 J1=1,2
 13          XDT(I1,J1)=I1+J1
 14      200 CONTINUE
 15      100 CONTINUE
 16          IF(A.EQ.B)
 17         $ X = CA1
 18          X = M+CA
 19          R = F101(1.)
 20          CALL SUB103(3,B+C,Y+1)
 21          CALL SUB103(A,B+C,Y+1)
 22          CALL SUB105(SUBEX,3)
 23          CALL SUB106(SUBFX,3)
 24          CALL SUB208(A,CA)
 25          DO 10 I = 1 , 10
 26          K = LOC + 1
 27       10 CONTINUE
 28          K = I + 6
 29          CALL SUB(1.,SUBEX)
 30          I = W201(CA)
 31          CALL SUB215(XDAT,B,C)
 32          CA = 1.
 33          CA = 2.
 34          BA=CA
 35          IF(D219.EQ.0)
 36         $ CA=3.+BA
 37          Y228(1)=6.+R
 38          X229=6.
 39          X230 = 1.
 40          IF(CA.EQ.L)
 41         $ X230=3.
 42          XDAT(5)=1.+X229+X230
 43          I = W201(X)+X


 44          I = W201(CA)+CA
 45          I = LASBF(2.,5.)+C
 46          I = FSIM(CA)
 47          CALL SUBSIM(X,A,B,D)
 48          STOP
  1          END
```

```
                                    E R R O R S
                                    - - - - - -
 ERROR
 NUMBER                            DESCRIPTION
 ------                            -----------

** 103 **  BLOCK NO.    19
           AN ACTUAL ARGUMENT IS AN EXPRESSION OR CONSTANT, YET THE
           CORRESPONDING DUMMY ARGUMENT IS ASSIGNED A VALUE ON ALL PATHS.
                                 CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                    -*SYSMAIN*-        ---*E101*--
                 ARGUMENT              REAL            -----*A*-----
                 POSITION               1                    1

** 103 **  BLOCK NO.    20
           AN ACTUAL ARGUMENT IS AN EXPRESSION OR CONSTANT, YET THE
           CORRESPONDING DUMMY ARGUMENT IS ASSIGNED A VALUE ON ALL PATHS.
                                 CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                    -*SYSMAIN*-        --*SUB103*-
                 ARGUMENT             INTEGER          -----*I*-----
                 POSITION               1                    1

** 103 **  BLOCK NO.    20
           AN ACTUAL ARGUMENT IS AN EXPRESSION OR CONSTANT, YET THE
           CORRESPONDING DUMMY ARGUMENT IS ASSIGNED A VALUE ON ALL PATHS.
                                 CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                    -*SYSMAIN*-        --*SUB103*-
                 ARGUMENT            EXPRESSION        -----*X*-----
                 POSITION               2                    2

** 103 **  BLOCK NO.    21
           AN ACTUAL ARGUMENT IS AN EXPRESSION OR CONSTANT, YET THE
           CORRESPONDING DUMMY ARGUMENT IS ASSIGNED A VALUE ON ALL PATHS.
                                 CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                    -*SYSMAIN*-        --*SUB103*-
                 ARGUMENT            EXPRESSION        -----*X*-----
                 POSITION               2                    2

** 103 **  BLOCK NO.    23
           AN ACTUAL ARGUMENT IS AN EXPRESSION OR CONSTANT, YET THE
           CORRESPONDING DUMMY ARGUMENT IS ASSIGNED A VALUE ON ALL PATHS.
                                 CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                    -*SYSMAIN*-        --*SUB106*-
                 ARGUMENT             INTEGER          -----*Z*-----
                 POSITION               2                    2
```

```
** 105 **  BLOCK NO.   22
           AN ACTUAL ARGUMENT IS A PROCEDURE DECLARED EXTERNAL, YET THE
           CORRESPONDING DUMMY ARGUMENT IS REFERENCED AS A VARIABLE
           ON ALL PATHS.
                                     CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                        -*SYSMAIN*-           --*SUB105*-
                     ARGUMENT           --*SUBEX*--           ----*X*----
                     POSITION                1                     1

** 106 **  BLOCK NO.   22
           AN ACTUAL ARGUMENT IS A PROCEDURE DECLARED EXTERNAL, YET THE
           CORRESPONDING DUMMY ARGUMENT, USED AS A VARIABLE, IS ASSIGNED
           A VALUE ON ALL PATHS.
                                     CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                        -*SYSMAIN*-           --*SUB105*-
                     ARGUMENT           --*SUBEX*--           ----*X*----
                     POSITION                1                     1

** 106 **  BLOCK NO.   23
           AN ACTUAL ARGUMENT IS A PROCEDURE DECLARED EXTERNAL, YET THE
           CORRESPONDING DUMMY ARGUMENT, USED AS A VARIABLE, IS ASSIGNED
           A VALUE ON ALL PATHS.
                                     CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                        -*SYSMAIN*-           --*SUB106*-
                     ARGUMENT           --*SUBEX*--           ----*X*----
                     POSITION                1                     1

** 108 **  BLOCK NO.   30
           A SUBPROGRAM REFERENCE CAUSES DUMMY ARGUMENT ----*X*----
           TO BECOME ASSOCIATED WITH A COMMON VARIABLE IN THE CALLED
           SUBPROGRAM.  ----*X*---- IS ASSIGNED A VALUE ON ALL PATHS.
                                     CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                        -*SYSMAIN*-           ---*W201*--
                     ARGUMENT           ----*CA*---           ----*X*----
           COMMON VARIABLE              ----*CA*---           ----*CA*---

** 108 **  BLOCK NO.   44
           A SUBPROGRAM REFERENCE CAUSES DUMMY ARGUMENT ----*X*----
           TO BECOME ASSOCIATED WITH A COMMON VARIABLE IN THE CALLED
           SUBPROGRAM.  ----*X*---- IS ASSIGNED A VALUE ON ALL PATHS.
                                     CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                        -*SYSMAIN*-           ---*W201*--
                     ARGUMENT           ----*CA*---           ----*X*----
           COMMON VARIABLE              ----*CA*---           ----*CA*---

** 109 **  COMMON VARIABLE ---*Y22B*-- IN COMMON BLOCK ---*BLK1*-- IS
           REFERENCED ON ALL PATHS IN THE MAIN PROGRAM, YET IT HAS NOT
           PREVIOUSLY BEEN ASSIGNED A VALUE, NOR HAS IT BEEN INITIALIZED
           IN BLOCK DATA.  (SEE NOTE 1)

** 109 **  COMMON VARIABLE ----*CA*--- IN COMMON BLOCK ---*BLK1*-- IS
           REFERENCED ON ALL PATHS IN THE MAIN PROGRAM, YET IT HAS NOT
           PREVIOUSLY BEEN ASSIGNED A VALUE, NOR HAS IT BEEN INITIALIZED
           IN BLOCK DATA.  (SEE NOTE 1)
```

```
** 110 **  COMMON VARIABLE ----*M*---- IS REFERENCED ON ALL PATHS IN


           CALLED SUBPROGRAM ---*F101*--, YET IS NOT INITIALIZED.  IT
           DOES NOT APPEAR IN BLOCK DATA, AND ITS COMMON BLOCK ---*F110*--
           IS NOT AVAILABLE TO CALLING SUBPROGRAM -*SYSMAIN*-.  (SEE
           NOTE 1)

** 110 **  COMMON VARIABLE ----*B*---- IS REFERENCED ON ALL PATHS IN
           CALLED SUBPROGRAM --*SUB103*-, YET IS NOT INITIALIZED.  IT
           DOES NOT APPEAR IN BLOCK DATA, AND ITS COMMON BLOCK ---*BLK*---
           IS NOT AVAILABLE TO CALLING SUBPROGRAM -*SYSMAIN*-.  (SEE
           NOTE 1)

** 110 **  COMMON VARIABLE ----*D*---- IS REFERENCED ON ALL PATHS IN
           CALLED SUBPROGRAM --*SUB208*-, YET IS NOT INITIALIZED.  IT
           DOES NOT APPEAR IN BLOCK DATA, AND ITS COMMON BLOCK ---*BLK*---
           IS NOT AVAILABLE TO CALLING SUBPROGRAM -*SYSMAIN*-.  (SEE
           NOTE 1)

** 111 **  CONTROL VARIABLE ----*I*---- BECOMES UNDEFINED UPON SATISFACTION
           OF ITS DO LOOP AT BLOCK NO.   27, YET IS REFERENCED ON ALL
           PATHS THEREAFTER.
           ONE SUCH PATH, INDICATED BY BLOCK NUMBERS, IS
             27   28

** 112 **  LOCAL VARIABLE ---*XDAT*-- IS REFERENCED BEFORE BEING ASSIGNED
           A VALUE ON ALL PATHS.
           ONE SUCH PATH, INDICATED BY BLOCK NUMBERS, IS
              1 -  31

** 112 **  LOCAL VARIABLE ----*A*---- IS REFERENCED BEFORE BEING ASSIGNED
           A VALUE ON ALL PATHS.
           ONE SUCH PATH, INDICATED BY BLOCK NUMBERS, IS
              1 -  16


                              W A R N I N G S
                              - - - - - - - -


  WARNING
  NUMBER                        DESCRIPTION
  ------                        -----------

** 203 **  BLOCK NO.   20
           AN ACTUAL ARGUMENT IS AN EXPRESSION OR CONSTANT, YET THE
           CORRESPONDING DUMMY ARGUMENT IS ASSIGNED A VALUE ON SOME PATHS.
                             CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                -*SYSMAIN*-         --*SUB103*-
                ARGUMENT         EXPRESSION          ----*Y*----
                POSITION             3                   3


** 203 **  BLOCK NO.   21
           AN ACTUAL ARGUMENT IS AN EXPRESSION OR CONSTANT, YET THE
```

```
          CORRESPONDING DUMMY ARGUMENT IS ASSIGNED A VALUE ON SOME PATHS.

                               CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                  -*SYSMAIN*-         --*SUB103*-
                 ARGUMENT          EXPRESSION          ----*Y*----
                 POSITION              3                    3

** 204 **  BLOCK NO.   19
           AN ACTUAL ARGUMENT IS AN EXPRESSION OR CONSTANT, YET THE
           CORRESPONDING DUMMY ARGUMENT IS NEVER REFERENCED.
                               CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                  -*SYSMAIN*-         ---*E101*--
                 ARGUMENT             REAL             ----*A*----
                 POSITION              1                    1

** 204 **  BLOCK NO.   20
           AN ACTUAL ARGUMENT IS AN EXPRESSION OR CONSTANT, YET THE
           CORRESPONDING DUMMY ARGUMENT IS NEVER REFERENCED.
                               CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                  -*SYSMAIN*-         --*SUB103*-
                 ARGUMENT           INTEGER            ----*I*----
                 POSITION              1                    1

** 204 **  BLOCK NO.   20
           AN ACTUAL ARGUMENT IS AN EXPRESSION OR CONSTANT, YET THE
           CORRESPONDING DUMMY ARGUMENT IS NEVER REFERENCED.
                               CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                  -*SYSMAIN*-         --*SUB103*-
                 ARGUMENT          EXPRESSION          ----*X*----
                 POSITION              2                    2

** 204 **  BLOCK NO.   21
           AN ACTUAL ARGUMENT IS AN EXPRESSION OR CONSTANT, YET THE
           CORRESPONDING DUMMY ARGUMENT IS NEVER REFERENCED.
                               CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                  -*SYSMAIN*-         --*SUB103*-
                 ARGUMENT          EXPRESSION          ----*X*----
                 POSITION              2                    2

** 204 **  BLOCK NO.   23
           AN ACTUAL ARGUMENT IS AN EXPRESSION OR CONSTANT, YET THE
           CORRESPONDING DUMMY ARGUMENT IS NEVER REFERENCED.
                               CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                  -*SYSMAIN*-         --*SUB106*-
                 ARGUMENT           INTEGER            ----*Z*----
                 POSITION              2                    2

** 205 **  BLOCK NO.   29
           AN ACTUAL ARGUMENT IS A PROCEDURE DECLARED EXTERNAL, YET THE
           CORRESPONDING DUMMY ARGUMENT IS REFERENCED AS A VARIABLE ON
           SOME PATHS.
                               CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                  -*SYSMAIN*-         ----*SUB*---
                 ARGUMENT         --*SUBFX*--          ----*B*----
                 POSITION              2                    2

** 206 **  BLOCK NO.   29
```

```
          AN ACTUAL ARGUMENT IS A PROCEDURE DECLARED EXTERNAL, YET THE


          CORRESPONDING DUMMY ARGUMENT, USED AS A VARIABLE, IS ASSIGNED
          A VALUE ON SOME PATHS.
                                  CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                     --*SYSMAIN*-         ---*SUB*---
                ARGUMENT             --*SUBEX*--          ----*B*----
                POSITION                  2                    2

** 208 **  BLOCK NO.    24
          A SUBPROGRAM REFERENCE CAUSES DUMMY ARGUMENT ----*X*----
          TO BECOME ASSOCIATED WITH A COMMON VARIABLE IN THE CALLED
          SUBPROGRAM.  ----*X*---- IS ASSIGNED A VALUE ON SOME PATHS.
                                  CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                     --*SYSMAIN*-         --*SUB208*-
                ARGUMENT             ----*CA*---          ----*X*----
          COMMON VARIABLE            ----*CA*---          ----*CA*---

** 209 **  COMMON VARIABLE ---*CA1*--- IN COMMON BLOCK ----*B1*--- IS
          REFERENCED ON SOME PATHS IN THE MAIN PROGRAM, YET IT HAS NOT
          PREVIOUSLY BEEN ASSIGNED A VALUE, NOR HAS IT BEEN INITIALIZED
          IN BLOCK DATA.  (SEE NOTE 1)


** 210 **  COMMON VARIABLE ----*C*---- IS REFERENCED ON SOME PATHS IN
          CALLED SUBPROGRAM --*SUB103*-, YET IS NOT INITIALIZED.
          IT DOES NOT APPEAR IN BLOCK DATA, AND ITS COMMON BLOCK
          ---*BLK*--- IS NOT AVAILABLE TO CALLING SUBPROGRAM
          -*SYSMAIN*-.  (SEE NOTE 1)


** 210 **  COMMON VARIABLE ----*D*---- IS REFERENCED ON SOME PATHS IN
          CALLED SUBPROGRAM --*SUB103*-, YET IS NOT INITIALIZED.
          IT DOES NOT APPEAR IN BLOCK DATA, AND ITS COMMON BLOCK
          ---*BLK*--- IS NOT AVAILABLE TO CALLING SUBPROGRAM
          -*SYSMAIN*-.  (SEE NOTE 1)

** 210 **  COMMON VARIABLE ----*B*---- IS REFERENCED ON SOME PATHS IN
          CALLED SUBPROGRAM --*SUB208*-, YET IS NOT INITIALIZED.
          IT DOES NOT APPEAR IN BLOCK DATA, AND ITS COMMON BLOCK
          ---*BLK*--- IS NOT AVAILABLE TO CALLING SUBPROGRAM
          -*SYSMAIN*-.  (SEE NOTE 1)

** 213 **  BLOCK NO.  21
          CORRESPONDING ARGUMENTS HAVE DIFFERENT DATA TYPES.
                                  CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                     --*SYSMAIN*-         --*SUB103*-
                ARGUMENT             ----*A*----          ----*I*----
                POSITION                  1                    1
                DATA TYPE               REAL                INTEGER

** 213 **  BLOCK NO.    23
          CORRESPONDING ARGUMENTS HAVE DIFFERENT DATA TYPES.
                                  CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                     --*SYSMAIN*-         --*SUB106*-
                ARGUMENT               INTEGER            ----*Z*----
                POSITION                  2                    2
                DATA TYPE              INTEGER               REAL
```

```
** 214 **  CORRESPONDING COMMON VARIABLES IN COMMON BLOCK ---*BLK1*--
           HAVE DIFFERENT DATA TYPES.
                                  CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                    -*SYSMAIN*-          --*SUB103*-
                 VARIABLE           ----*CA*---          ----*K*----
                 DATA TYPE             REAL               INTEGER


** 214 **  CORRESPONDING COMMON VARIABLES IN COMMON BLOCK ---*BLK1*--
           HAVE DIFFERENT DATA TYPES.
                                  CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                    -*SYSMAIN*-          --*SUB103*-
                 VARIABLE           ---*D218*--          ----*K*----
                 DATA TYPE             REAL               INTEGER


** 214 **  CORRESPONDING COMMON VARIABLES IN COMMON BLOCK ---*BLK1*--
           HAVE DIFFERENT DATA TYPES.
                                  CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                    -*SYSMAIN*-          --*SUB103*-
                 VARIABLE           ---*Y228*--          ----*K*----
                 DATA TYPE             REAL               INTEGER

** 215 **  BLOCK NO.   31
           CORRESPONDING ARGUMENTS HAVE DIFFERENT DIMENSIONALITY.
                                  CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                    -*SYSMAIN*-          --*SUB215*-
                 ARGUMENT           ---*XDAT*--         ---*XDAT*--
                 POSITION                1                   1
                 DIMENSIONS              1                   2

** 216 **  COMMON VARIABLE ----*N*---- IS ASSIGNED A VALUE ON ALL PATHS
           IN CALLED SUBPROGRAM ---*F101*--, YET ITS COMMON BLOCK
           ---*E110*-- IS NOT AVAILABLE TO CALLING SUBPROGRAM -*SYSMAIN*-.
           HENCE, A COMPUTED VALUE WILL BE LOST.  (SEE NOTE 1)


** 216 **  COMMON VARIABLE ----*B*---- IS ASSIGNED A VALUE ON ALL PATHS
           IN CALLED SUBPROGRAM --*SUB103*-, YET ITS COMMON BLOCK
           ---*BLK*--- IS NOT AVAILABLE TO CALLING SUBPROGRAM -*SYSMAIN*-.
           HENCE, A COMPUTED VALUE WILL BE LOST.  (SEE NOTE 1)

** 216 **  COMMON VARIABLE ----*D*---- IS ASSIGNED A VALUE ON ALL PATHS
           IN CALLED SUBPROGRAM --*SUB103*-, YET ITS COMMON BLOCK
           ---*BLK*--- IS NOT AVAILABLE TO CALLING SUBPROGRAM -*SYSMAIN*-.
           HENCE, A COMPUTED VALUE WILL BE LOST.  (SEE NOTE 1)


** 216 **  COMMON VARIABLE ----*C*---- IS ASSIGNED A VALUE ON ALL PATHS
           IN CALLED SUBPROGRAM --*SUB215*-, YET ITS COMMON BLOCK
           ---*BLK*--- IS NOT AVAILABLE TO CALLING SUBPROGRAM -*SYSMAIN*-.
           HENCE, A COMPUTED VALUE WILL BE LOST.  (SEE NOTE 1)

** 217 **  COMMON VARIABLE ----*C*---- IS ASSIGNED A VALUE ON SOME PATHS
           IN CALLED SUBPROGRAM --*SUB103*-, YET ITS COMMON BLOCK
           ---*BLK*--- IS NOT AVAILABLE TO CALLING SUBPROGRAM
           -*SYSMAIN*-.  HENCE, A COMPUTED VALUE MAY BE LOST.  (SEE
           NOTE 1)
```

```
** 217 **  COMMON VARIABLE ----*D*---- IS ASSIGNED A VALUE ON SOME PATHS
           IN CALLED SUBPROGRAM --*SUB215*-, YET ITS COMMON BLOCK
           ---*BLK*--- IS NOT AVAILABLE TO CALLING SUBPROGRAM
           -*SYSMAIN*-,  HENCE, A COMPUTED VALUE MAY BE LOST.  (SEE
           NOTE 1)

** 218 **  COMMON VARIABLE ----*T*---- IS INITIALIZED IN BLOCK DATA.
           IT IS ASSIGNED A VALUE ON ALL PATHS IN CALLED SUBPROGRAM
           --*SUB215*-, YET ITS COMMON BLOCK ---*IBD*--- IS NOT AVAILABLE
           TO CALLING SUBPROGRAM -*SYSMAIN*-.  HENCE, UNDEFINITION WILL
           OCCUR UPON EXIT FROM --*SUB215*-. (SEE NOTE 2)

** 219 **  COMMON VARIABLE ----*W*---- IS INITIALIZED IN BLOCK DATA.
           IT IS ASSIGNED A VALUE ON SOME PATHS IN CALLED SUBPROGRAM
           --*SUB215*-, YET ITS COMMON BLOCK ---*IBD*--- IS NOT AVAILABLE
           TO CALLING SUBPROGRAM -*SYSMAIN*-.  HENCE, UNDEFINITION MAY
           OCCUR UPON EXIT FROM --*SUB215*-. (SEE NOTE 2)

** 226 **  IN THE MAIN PROGRAM, COMMON VARIABLE ----*CA*--- IS
           ASSIGNED A VALUE IN BLOCK  NO.   32 AND IS EITHER
           ASSIGNED A VALUE THEREAFTER BEFORE BEING REFERENCED,
           OR IS NOT SUBSEQUENTLY REFERENCED, ON ALL PATHS.
           ONE SUCH PATH, INDICATED BY BLOCK NUMBERS, IS
              32    33

** 227 **  IN THE MAIN PROGRAM, COMMON VARIABLE ----*HA*--- IS
           ASSIGNED A VALUE IN BLOCK  NO.   34 AND IS EITHER
           ASSIGNED A VALUE THEREAFTER BEFORE BEING REFERENCED,
           OR IS NOT SUBSEQUENTLY REFERENCED, ON SOME PATHS.
           ONE SUCH PATH, INDICATED BY BLOCK NUMBERS, IS
              34    35   37 -  48

** 228 **  IN THE MAIN PROGRAM, AN ELEMENT OF THE COMMON ARRAY
           ---*Y228*-- IS ASSIGNED A VALUE IN BLOCK NO.   37
           AND THE ARRAY IS NOT SUBSEQUENTLY REFERENCED ON ANY PATH.

** 229 **  LOCAL VARIABLE ---*I236*-- IS ASSIGNED A VALUE IN BLOCK
           NO.    5 AND IS EITHER ASSIGNED A VALUE THEREAFTER BEFORE
           BEING REFERENCED, OR IS NOT SUBSEQUENTLY REFERENCED,
           ON ALL PATHS.
           ONE SUCH PATH, INDICATED BY BLOCK NUMBERS, IS
               5 -  48

** 229 **  LOCAL VARIABLE ----*X*---- IS ASSIGNED A VALUE IN BLOCK
           NO.   17 AND IS EITHER ASSIGNED A VALUE THEREAFTER BEFORE
           BEING REFERENCED, OR IS NOT SUBSEQUENTLY REFERENCED,
           ON ALL PATHS.
           ONE SUCH PATH, INDICATED BY BLOCK NUMBERS, IS
              17    18

** 229 **  LOCAL VARIABLE ----*K*---- IS ASSIGNED A VALUE IN BLOCK
           NO.   26 AND IS EITHER ASSIGNED A VALUE THEREAFTER BEFORE
           BEING REFERENCED, OR IS NOT SUBSEQUENTLY REFERENCED,
          ON ALL PATHS.

          ONE SUCH PATH, INDICATED BY BLOCK NUMBERS, IS
            26    27    28

** 230 ** LOCAL VARIABLE ---*X230*-- IS ASSIGNED A VALUE IN BLOCK
          NO.   39 AND IS EITHER ASSIGNED A VALUE THEREAFTER BEFORE
          BEING REFERENCED, OR IS NOT SUBSEQUENTLY REFERENCED,
          ON SOME PATHS.
          ONE SUCH PATH, INDICATED BY BLOCK NUMBERS, IS
            39    40    41

** 231 ** AN ELEMENT OF THE LOCAL ARRAY ---*XDAT*-- IS ASSIGNED A VALUE
          IN BLOCK NO.   42 AND THE ARRAY IS NOT SUBSEQUENTLY
          REFERENCED ON ANY PATH.

** 231 ** AN ELEMENT OF THE LOCAL ARRAY ---*XDT*--- IS ASSIGNED A VALUE
          IN BLOCK NO.   13 AND THE ARRAY IS NOT SUBSEQUENTLY
          REFERENCED ON ANY PATH.


** 232 ** BLOCK NO.   43
          A POSSIBLE ILLEGAL SIDE EFFECT HAS BEEN DETECTED.  IT OCCURS
          VIA A VARIABLE PASSED IN AN ARGUMENT LIST.  THIS VARIABLE
          HAS APPEARED AT LEAST TWICE IN A STATEMENT -- IN ONE
          APPEARANCE IT IS USED AS STRICT INPUT AND IN THE OTHER AS
          STRICT OUTPUT.
                             CALLING SUBPROGRAM  CALLED SUBPROGRAM
                               -*SYSMAIN*-           ---*W201*--
                ARGUMENT       ----*X*----           ----*X*----
                POSITION            1                     1

** 232 ** BLOCK NO.   44
          A POSSIBLE ILLEGAL SIDE EFFECT HAS BEEN DETECTED.  IT OCCURS
          VIA A VARIABLE PASSED IN AN ARGUMENT LIST.  THIS VARIABLE
          HAS APPEARED AT LEAST TWICE IN A STATEMENT -- IN ONE
          APPEARANCE IT IS USED AS STRICT INPUT AND IN THE OTHER AS
          STRICT OUTPUT.
                             CALLING SUBPROGRAM  CALLED SUBPROGRAM
                               -*SYSMAIN*-           ---*W201*--
                ARGUMENT       ----*CA*---           ----*X*----
                POSITION            1                     1

** 233 ** BLOCK NO.   30
          A POSSIBLE ILLEGAL SIDE EFFECT HAS BEEN DETECTED.  IT OCCURS
          VIA A COMMON VARIABLE WHICH HAS BEEN REFERENCED (POSSIBLY
          INDIRECTLY) AT LEAST TWICE IN A STATEMENT -- IN ONE APPEAR-
          ANCE IT IS USED AS STRICT INPUT AND IN THE OTHER AS STRICT
          OUTPUT.
                             CALLING SUBPROGRAM  CALLED SUBPROGRAM
                               -*SYSMAIN*-           ---*W201*--
                VARIABLE       ----*CA*---           ----*CA*---
            COMMON BLOCK       ---*BLK1*--           ---*BLK1*--

** 233 ** BLOCK NO.   44
          A POSSIBLE ILLEGAL SIDE EFFECT HAS BEEN DETECTED.  IT OCCURS
          VIA A COMMON VARIABLE WHICH HAS BEEN REFERENCED (POSSIBLY
          INDIRECTLY) AT LEAST TWICE IN A STATEMENT -- IN ONE APPEAR-
```

```
         ANCE IT IS USED AS STRICT INPUT AND IN THE OTHER AS STRICT

         OUTPUT.
                               CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                  -*SYSMAIN*-         ---*W201*--
                 VARIABLE         -----*CA*---        -----*CA*---
             COMMON BLOCK         ---*BLK1*--         ---*BLK1*--

** 234 **  BLOCK NO.    45
           A POSSIBLE ILLEGAL SIDE EFFECT HAS BEEN DETECTED.  IT OCCURS
           VIA A GLOBAL VARIABLE REFERENCED IN AN ARITHMETIC STATEMENT
           FUNCTION.  THIS VARIABLE HAS APPEARED AT LEAST TWICE IN A
           STATEMENT -- IN ONE APPEARANCE IT IS USED AS STRICT INPUT AND
           IN THE OTHER AS STRICT OUTPUT.
                               CALLING SUBPROGRAM  CALLED SUBPROGRAM
                                  -*SYSMAIN*-         --*LASHF*--
                 VARIABLE         ----*C*----         --*       *-


                                 M E S S A G E S
                                 - - - - - - - -

  MESSAGE
  NUMBER                        DESCRIPTION
  ------                        -----------


** 301 **  COMMON VARIABLE ---*D218*-- IN BLOCK ---*BLK1*-- OF
           SUBPROGRAM -*SYSMAIN*- IS INITIALIZED IN BLOCK DATA.


** 301 **  COMMON VARIABLE ----*W*---- IN BLOCK ---*IBD*--- OF
           SUBPROGRAM --*SUB215*- IS INITIALIZED IN BLOCK DATA.

** 303 ** THE FOLLOWING DATA FLOW OCCURS THROUGH COMMON WHEN SUBPROGRAM
          --*SUB103*- IS CALLED.

             COMMON                          INPUT           OUTPUT
              BLOCK        VARIABLE     CLASSIFICATION  CLASSIFICATION
              -----        --------     --------------  --------------

           ---*BLK1*--  -----*CA*---        STRICT           NON
           ---*BLK1*--  ---*D218*--         STRICT           NON
           ---*BLK1*--  ---*Y228*--         STRICT           NON

** 303 ** THE FOLLOWING DATA FLOW OCCURS THROUGH COMMON WHEN SUBPROGRAM
          ---*W201*-- IS CALLED.

             COMMON                          INPUT           OUTPUT
              BLOCK        VARIABLE     CLASSIFICATION  CLASSIFICATION
              -----        --------     --------------  --------------

           ---*BLK1*--  -----*CA*---        STRICT           NON

** 304 ** I/O CLASSIFICATION OF ARGUMENTS AND COMMON VARIABLES
          FOR -*SYSMAIN*-
          COMMON BLOCK       ----*H1*---

               AVAILABILITY = ORIGINAL

          ARGUMENTS
               POSITION      NAME      INPUT CLASS      OUTPUT CLASS

                  1       ---*CA1*---      INPUT             NON
                  2       ----*BA*---       NON             STRICT

          COMMON BLOCK       ---*BLK1*--

               AVAILABILITY = ORIGINAL

          ARGUMENTS
               POSITION      NAME      INPUT CLASS      OUTPUT CLASS

                  1       ----*CA*---     STRICT            STRICT
                  2       ---*D21A*--     STRICT             NON
                  3       ---*Y22H*--     STRICT            STRICT

          : :                --*LASHF*--

          ARGUMENTS
               POSITION      NAME      INPUT CLASS      OUTPUT CLASS

                  1       ----*X*----     STRICT             NON
                  2       ----*Y*----     STRICT             NON


                                  NOTES
                                  -----
NOTE 1    ALTHOUGH DETECTED IN THIS SUBPROGRAM, THE CAUSE FOR THIS
---- -    DIAGNOSTIC MAY HAVE OCCURRED AT A DEEPER LEVEL OF SUBPROGRAM
          REFERENCES AND BEEN PROPAGATED UP TO THIS ONE.

NOTE 2    IF MESSAGE 301 CONCERNING THIS VARIABLE APPEARS IN THE
---- -    OUTPUT, IT MAY PROVIDE ADDITIONAL USEFUL INFORMATION
          ABOUT THE DATA FLOW AMONG SUBPROGRAMS.
```

DISTRIBUTION LIST

No. of Copies	Organization
12	Commander Defense Documentation Center ATTN: DDC-TCA Cameron Station Alexandria, VA 22314
1	Commander US Army Materiel Development and Readiness Command ATTN: DRCDMD-ST 5001 Eisenhower Avenue Alexandria, VA 22333
1	Commander US Army Aviation Research and Development Command ATTN: DRSAV-E 12th and Spruce Streets St. Louis, MO 63166
1	Director US Army Air Mobility Research and Development Laboratory Ames Research Center Moffett Field, CA 94035
1	Commander US Army Electronics Research and Development Command Technical Support Activity ATTN: DELSD-L Fort Monmouth, NJ 07703
1	Commander US Army Communications Rsch and Development Command ATTN: DRDCO-SGS Fort Monmouth, NJ 07703
1	Commander US Army Missile Research and Development Command ATTN: DRDMI-R Redstone Arsenal, AL 35809
1	Commander US Army Missile Materiel Readiness Command ATTN: DRSMI-AOM Redstone Arsenal, AL 35809
1	Commander US Army Tank Automotive Research & Development Cmd ATTN: DRDTA-UL Warren, MI 48090
1	Commander US Army Armament Materiel Readiness Command ATTN: DRSAR-LEP-L, Tech Lib Rock Island, IL 61299
4	Commander US Army Armament Research and Development Command ATTN: DRDAR-TSS (2 cys) DRDAR-MS, D. Grobstein DRDAR-MSE, S.A.Goldberg Dover, NJ 07801
1	Director US Army TRADOC Systems Analysis Activity ATTN: ATAA-SL, Tech Lib White Sands Missile Range NM 88002
2	Commander US Army BMD Advanced Technology Center ATTN: ATC-P, Mr. C. Vick Dr. Carl Davis P. O. Box 1500 Huntsville, AL 35807
1	AFSC/ASD (Dr. Richard Sylvester) Wright-Patterson AFB, OH 45433

PRECEDING PAGE BLANK-NOT FILMED

DISTRIBUTION LIST

No. of Copies	Organization
1	Director National Aeronautics and Space Administration Langley Research Center ATTN: Dr. T. Straeter Mail Stop 125A Hampton, VA 23665
2	Boeing Computer Science Co. ATTN: Dr. L. Stucki Mr. J. Brown Seattle, WA 98124
1	Flow General Corporation ATTN: Sabina Saib P. O. Box 3587 Santa Barbara, CA 93105
1	SOFTECH, Inc. ATTN: Dr. J. Goodenough 460 Totten Pond Road Waltham, MA 02154
1	Software Research Associates ATTN: Dr. E. F. Miller, Jr. P. O. Box 2432 San Francisco, CA 94126
2	SRI International ATTN: Dr. B. Elspas Dr. J. Goldberg 333 Ravenswood Avenue Menlo Park, CA 94025
2	University of Colorado Dept of Computer Science ATTN: Dr. L. Fosdick Dr. L. Osterweil Boulder, CO 80309
2	University of Delaware ATTN: Dr. H. Khalil Dr. L. Levy Newark, DE 19711
1	The Johns Hopkins University ATTN: Dr. John H. Manley Baltimore, MD 21234
1	University of Maryland at College Park Dept of Computer Science ATTN: Dr. V. R. Basili Dr. M. V. Zelkowitz College Park, MD 20742
1	University of Massachusetts Dept of Computer and Information Science ATTN: Dr. L. Clarke Amherst, MA 01002
1	University of North Carolina at Chapel Hill Department of Computer Science ATTN: Dr. D. Parnas Chapel Hill, NC 27514
1	University of Texas at Austin Department of Computer Science and Computation Center ATTN: Dr. J. C. Browne Austin, TX 78712
1	Virginia Polytechnic and State University Department of Computer Science ATTN: B. G. Claybrook Blacksburg, VA 24061

Aberdeen Proving Ground

Cdr, USATECOM
ATTN: Mr. Barnhart
DRSTE-SG-H

(continued on next page)

DISTRIBUTION LIST

Aberdeen Proving Ground (continued)

Dir, USAMSAA
ATTN: Dr. Sperrazza
Dr. Atzinger
Mr. L. Bain
Mr. J. Bannon
Mr. E. Belbot
MAJ Cecil
Mr. Clay
Dr. Cooper
Dr. Cummings
Mr. Daily
Mr. Dombeck
Dr. Dubin
Mr. Holloway
Mr. Iten
Mr. Kilminster
Mr. Lufkin
Mr. McCarthy
Mr. Sebra
Dr. Streilein
Mr. Towson
Mr. Warfield